A short note on the tractability of constructing phylogenetic networks from clusters

Leo van Iersel[1], Steven Kelk[2]

[1] University of Canterbury, Department of Mathematics and Statistics, Private Bag 4800, Christchurch, New Zealand, email: `l.j.j.v.iersel@gmail.com`.
[2] Centrum voor Wiskunde en Informatica (CWI), Life Sciences, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, email: `s.m.kelk@cwi.nl`.

Abstract In [2] it was proven that the CASS algorithm is a polynomial-time algorithm for constructing level-≤ 2 networks from clusters[3]. Here we demonstrate, for each $k \geq 0$, a polynomial-time algorithm for constructing level-k phylogenetic networks from clusters. Unlike CASS the algorithm scheme given here is only of theoretical interest. It does, however, strengthen the hope that efficient polynomial-time algorithms (and perhaps fixed parameter tractable algorithms) exist for this problem.

We refer the reader to [2] for definitions of the terminology used throughout this note.

Theorem 1. *Let C be a set of clusters on a taxa set X on n leaves. Then, for every fixed $k \geq 0$, it is possible to determine in polynomial time whether a level-k network exists that represents C, and if so to construct such a network.*

Proof. Suppose C has the property that for every $X' \subset X$, X' is separated. We call such a cluster set C *fully separated*. It was shown in [2] that the existence of a polynomial-time algorithm for constructing a level-k network from a fully separated cluster set, is sufficient to give a polynomial-time algorithm for constructing level-k networks from general cluster sets. (Specifically, the fully-separated cluster sets are obtained by processing each non-trivial connected component of the *incompatibility graph* of the original cluster set [2]). Hence we can assume without loss of generality that C is fully separated. Furthermore, it was also shown in [2] that (i) any network that represents C is simple, and (ii) if there exists a level-k network that represents C, then there exists a binary simple level-k network N that represents C [2]. Lemma 1 is thus sufficient, and we are done. □

Lemma 1. *Let C be a fully separated cluster set on a taxa set X, where $|X| = n$. Then, for every fixed $k \geq 0$, it is possible to determine in polynomial time whether a binary simple level-k network exists that represents C, and if so to construct such a network.*

---

[3] In this note we are referring exclusively to *softwired* clusters, as opposed to hardwired clusters.

Proof. We assume that k is fixed. Assume then that there does exist a binary simple level-k network N that represents C. Then C will contain at most $2^{k+1}(n-1)$ clusters, because there are at most 2^k trees displayed by a binary simple level-k network, and each tree represents at most $2(n-1)$ clusters. Thus, for fixed k, the size of the input is polynomial in n. Note also that because of these facts it is easy to check in polynomial time whether a set of clusters is indeed represented by a given simple level-k network. In the remainder of this proof we will use this fact implicitly to distinguish correct from incorrect guesses.

It is known that, if the leaves of N are removed and all vertices with both indegree and outdegree equal to 1 are suppressed, the resulting structure will be a level-k generator, defined in [1]. For fixed k, there are only a constant number of level-k generators. Recall that the *sides* of a level-k generator are defined as the union of its edges and its vertices of indegree-2 and outdegree-0. For fixed k the maximum number of sides ranging over all level-k generators, is a constant.

For a cluster set C on X, we write $x \to y$ iff every non-singleton cluster in C that contains x, also contains y. Let $G(C) = (X, E)$ be a directed graph on X with edge set as follows: $(x, y) \in E$ iff $x \to y$ and there is no $z \notin \{x, y\}$ such that $x \to z$ and $z \to y$. It is easy to see that $G(C)$, which we call the *containment graph*, is acyclic: the presence of cycle in $G(C)$ would mean that all subsets of the taxa in the cycle are unseparated, contradicting the assumption of the lemma.

We propose the following simple algorithm for determining whether C is represented by a binary simple level-k network. In particular, we will attempt to reconstruct N. Let g be the generator underlying N. We only require polynomially many "guesses" to compute g, because there are only a constant number of generators. So assume we know g. For each side of g, we guess whether there are 0, 1, 2 or more than 2 leaves on that side. For each side containing exactly one leaf, we guess what that is. For each side s of g containing 2 or more leaves, we guess the leaf s^+ that is nearest to the root on that side, and the leaf s^- that is furthest from the root on that side.

We will now show how to add the remaining leaves. The critical point to note is that every remaining leaf will be added between the $+$ and the $-$ leaf on some side. We add the remaining leaves in a specific order. In particular, we say that a side s is *lowest* if it does not yet have all its leaves, and there is no other such side s' reachable from s. By reachable we mean: in the underlying generator g, there is a directed path from the head of side s to the tail of side s'. (The sides for which we guessed that they have 0, 1 or exactly 2 leaves, can never be lowest). N is a directed acyclic graph, so until all remaining leaves have been added, there will always be a lowest side.

The idea is to add leaves to the lowest side s, until all its leaves have been added. We then continue with remaining lowest sides until we have reconstructed N.

It is possible to tell in polynomial time what the correct leaves are for that side, as follows. Observe that a leaf x that is on side s in N has the property

$s^+ \to x \to s^-$. (Clearly $x \not\to s^+$ and $s^- \not\to x$). Furthermore, there is at least one cluster $c \in C$ such that $\{x, s^+, s^-\} \cap c = \{x, s^-\}$. We call such a cluster a *split cluster for side s*. There exists at least one such cluster because otherwise $\{x, s^+\}$ would be unseparated in C. Now, observe that for every split cluster c for side s, and for every side $t \neq s$ that contains 2 or more leaves in N, either $\{t^+, t^-\} \cap c = \{t^+, t^-\}$ or $\{t^+, t^-\} \cap c = \emptyset$. This follows because the only edges in N that represent c lie on side s. Now, consider any leaf y that has not yet been added to the network and is not on side s in N, but side t (for some t). Side t will contain three or more leaves in N, so we can assume that t^+ and t^- exist. If it is not the case that $s^+ \to y \to s^-$ then it is immediately clear that y cannot be put on side s. So assume (conversely) this condition does hold, and for the same reason assume there is a split cluster c for side s that contains y. In other words, there is a cluster c such that $\{y, s^+, s^-\} \cap c = \{y, s^-\}$. It follows that c also contains t^+ and t^-, because (by inspection on N) any cluster that contains y also contains t^-, and we know that c contains either both of t^+ and t^-, or neither of them. However, there is no edge in N that can represent c: the only edges that represent c lie on side s, but the fact that s is the lowest side means that no cluster beginning on side s can contain any leaves on side t. To summarise, then, we have a simple test for determining whether a leaf should be placed on side s. Once we have determined the set of leaves that should be placed on side s, it is easy to determine the correct order of those leaves by inspecting the containment graph. This concludes the proof. □

It is interesting to note that the above proof technique leads to a simplified proof, presented in the following Corollary, of a result that was first proven in [3]. (The algorithm presented in [3] was, however, far more efficient). We refer the reader to [1][3] for definitions related to triplets.

Corollary 1. *Let T be a dense set of triplets on leaf set L on n leaves. Then, for every fixed $k \geq 0$, it is possible to determine in polynomial time whether a binary simple level-k network exists that is consistent with T, and if so to construct such a network.*

Proof. The proof of Lemma 1 holds here almost entirely. (As noted in [3] it is possible to determine in polynomial time whether a given network is indeed consistent with a set of input triplets). The only significant difference concerns the adding of leaves to the lowest side. The crucial fact here is that a not yet allocated leaf x belongs on lowest side s if and only if the triplet $s^- x | s^+$ is in the input. □

References

1. L. J. J. van Iersel, J. C. M. Keijsper, S. M. Kelk, L. Stougie, F. Hagen, and T. Boekhout, Constructing level-2 phylogenetic networks from triplets, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 6(4), pp. 667–681, October 2009.

2. L. J. J. van Iersel, S. M. Kelk, R. Rupp and D. Huson, Phylogenetic Networks Do not Need to Be Complex: Using Fewer Reticulations to Represent Conflicting Clusters, arXiv:0910.3082v1 [q-bio.PE], October 2009.
3. L. J. J. van Iersel and S. M. Kelk. Constructing the simplest possible phylogenetic network from triplets, *Algorithmica*, 2009. To appear.